

# Design of Novel Architectures and FPGA Implementation of 2D Gaussian Surround Function

M. C Hanumantharaju and M. T Gopalakrishna

Department of Information Science and Engineering, Dayananda Sagar College of Engineering,  
Bangalore 560 078, India, Contact: mchanumantharaju@gmail.com, gopalmtm@gmail.com

A new design and novel architecture suitable for FPGA/ASIC implementation of a 2D Gaussian surround function for image processing application is presented in this paper. The proposed scheme results in enormous savings of memory normally required for 2D Gaussian function implementation. In the present work, the Gaussian symmetric characteristics which quickly falls off toward plus/minus infinity has been used in order to save the memory. The 2D Gaussian function implementation is presented for use in applications such as image enhancement, smoothing, edge detection and filtering etc. The FPGA implementation of the proposed 2D Gaussian function is capable of processing (blurring, smoothing, and convolution) high resolution color pictures of size up to  $1600 \times 1200$  pixels at the real time video rate of 30 frames/sec. The Gaussian design exploited here has been used in the core part of retinex based color image enhancement. Therefore, the design presented produces Gaussian output with three different scales, namely, 16, 64 and 128. The design was coded in Verilog, a popular hardware design language used in industries, conforming to RTL coding guidelines and fits onto a single chip with a gate count utilization of 89,213 gates. Experimental results presented confirms that the proposed method offers a new approach for development of large sized Gaussian pyramid while reducing the on-chip memory utilization.

**Keywords :** Gaussian Surround Function, Hardware Architecture, FPGA, Verilog.

## 1. Introduction

With the widespread use of technologies like digital television, internet streaming video and DVD video, Gaussian function has become an inevitable component of image/video processing [10] and pattern recognition. Hardware realization of 2D Gaussian surround function for image processing applications demands huge on-chip memory requirement, with massive computations. Further, these functions might not fit on a single FPGA device. This is due to the reason that Gaussian function has an exponential distribution with maximum entropy and implementation such functions using software schemes are complex from computation point of view. In addition, Gaussian function is a non-causal, which implies that function is symmetric about the origin in time domain. Gaussian filter implementation with smaller kernel size in order to blur an image have been reported by number of researchers [8].

However, design of Gaussian function for large kernel size requires enormous amount of resources on FPGA with incredible raise in the total equivalent gate count.

The new approach for 2D Gaussian function design provides a technical solution appropriate for the broad range of applications, from image pre-processing to pattern recognition. It simplifies the computational complexity with considerable saving in the memory requirement. The work proposed here utilizes the symmetry property of Gaussian surround function in order to save hardware resource, particularly memory requirement on FPGA. The technical design presented in this work is highly focused on providing huge throughput to process images as well as motion pictures. The Gaussian surround function is designed to process high quality images with picture size exceeding  $1600 \times 1200$  using reduced memory and high speed computation.

The rest of the paper is organized as follows:

Section 2 presents the review of related work. Section 3 describes the proposed 2D Gaussian surround function design. Section 4 gives brief details of architecture development suitable for FPGA/ASIC implementation. Section 5 provides experimental results and discussions. Finally conclusion arrived at is presented in Section 6.

## 2. Related Work

Image and Video Processing has been a very active field of research and development for over 20 years and many different systems and algorithms for image enhancement, restoration, filtering and smoothing have been proposed and developed. The core part in all these image processing techniques is the Gaussian function. Although a lot of research work has progressed in the development of Gaussian design for image processing application using software and hardware schemes, very little work seems to have been carried out in large sized Gaussian function design. In addition, there are no full-fledged implementations reported for 2D Gaussian surround function using FPGA or ASIC that demands the development of novel algorithms for high speed processing and the best possible memory utilization. The market demands for 2D Gaussian function design are very high. The design of 2D Gaussian function and architecture development [7] is not only challenging but also intellectually stimulating. These are the primary reasons why this work was undertaken by the present researcher.

Jobson et al. [1] described the properties of center or surround retinex. In this work authors have claimed that the design of a Gaussian surround function with a space constant of 80 pixels is a reasonable compromise between dynamic range and rendition. Gaussian function design and its application to retinex based image enhancement with DSP implementation has been proposed by Glenn et al. [2]. DSPs can be employed for enhancement of images which provides some improvement compared to general purpose computers. DSPs can be programmed in different languages, such as assembly codes and C language. The Hardware knowledge is required for programming DSPs, However, it is much easier

for designers to learn DSP programming compared with the other design choices. However, image enhancement algorithms involving Gaussian function developed for DSP implementation may not be parallelized without using multiple DSPs. The DSPs are well suited for implementation of floating point systems, while for ASICs and FPGAs, floating point operations are difficult to implement. The enhancement of 25-30 frames per second of large size video frames with  $1024 \times 1024$  pixel resolution is still not possible with DSPs. The image enhancement technique requires massive parallel processing capability in order to support real time enhancement of large video stream.

Hiroshi et al. [4] proposed a real time retinex video image enhancement algorithm and its FPGA implementation. Although the retinex algorithm [9] has been used for video enhancement the Gaussian pyramid designed in this work is of size  $3 \times 3$ . The authors have claimed that the architectures developed in this scheme are efficient and can handle color picture of size  $1900 \times 1200$  pixels at the real time video rate of 60 frames per sec. In this scheme, the computational cost of the algorithm depends on the number of processing layers while the maximum layers and iterations used are 5 and 30 respectively. The authors have not justified how high throughput has been achieved in spite of time consuming iterations to the tune of 30. Further, the algorithm uses HSV color space for the enhancement process. This leads to an additional computational cost with the maximum conversion error occurring in the conversion process from RGB to HSV.

Tsung et al. [5] proposed algorithm and architecture design of human machine interaction in foreground detection of dynamic scene. The Gaussian function in this work is computed based on fixed point data. This method adapts 27 bits for the variance computation of which 7 bits represents integer portion and 20 bits are used for fraction part. Three look-up tables are used to index the exponential and divide value. However, the Gaussian density function developed here uses large memory and is not efficient from computational point of view.

Design of novel algorithm and architecture for

Gaussian based color image enhancement for real time applications have been proposed by Hanumantharaju et al. [3]. The  $5 \times 5$  Gaussian kernel exploited in this work performs the image enhancement operation efficiently. Although the medium sized kernel employed in this work operates on picture size of  $1600 \times 1200$  pixels based on window operation, resulting images are not satisfactory due to presence of halo artifacts in the reconstructed images. Design of large sized Gaussian function in place of  $5 \times 5$  kernel improves visual quality of the enhanced image. Further, this approach consumes enormous amount of FPGA resources. The proposed approach of Gaussian surround function design with reduced on-chip memory utilization is an ideal choice compared to kernel based implementations. In addition, design of large size Gaussian surround function and development of architecture suitable for FPGA/ASIC implementation is the first-of-its-kind in the literature.

### 3. Design of 2D Gaussian Surround Function

The analog Gaussian function in 1D, 2D and ND is expressed as follows:

$$G_{1D}(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} \quad (1)$$

$$G_{2D}(x, y) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (2)$$

$$G_{ND}(\vec{x}) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{|\vec{x}|^2}{2\sigma^2}} \quad (3)$$

$\sigma$  indicates the width of the Gaussian function. According to statistics, if Gaussian probability density function is considered, than  $\sigma$  is referred to as standard deviation and the square of it ( $\sigma^2$ ) is called variance.

The discrete version of 2D Gaussian surround function is given by Eqn. (4).

$$G_n(x, y) = K_n \times e^{-\frac{(x^2+y^2)}{2\sigma^2}} \quad (4)$$

and  $K_n$  is given by the Eqn. (5)

$$K_n = \frac{1}{\sum_{i=1}^M \sum_{j=1}^N e^{-\frac{(x^2+y^2)}{2\sigma^2}}} \quad (5)$$

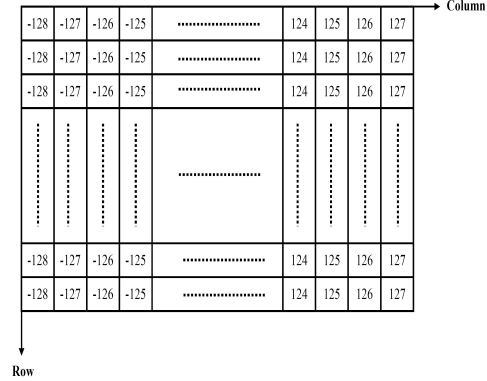


Figure 1. Domain Specified by 'x' Vector Transformed into Array 'x'

where  $x$  and  $y$  signify the spatial coordinates,  $M \times N$  represents the kernel size,  $n$  is preferred as 1, 2 and 3 since the Gaussian function is designed with three scales, namely, 16, 64, and 128.

The spatial co-ordinate 'x' shown in Eqn. (4) are derived from the vector  $x$  which is presented in Figure 1. The spatial co-ordinate 'y' is obtained similar to that of co-ordinate 'x'. It may be observed from the Figures 1 and 2 that the co-ordinates of Gaussian surround function exhibit symmetry property around the origin. Therefore, the spatial co-ordinate for implementation of Gaussian surround functions can be easily stored in the memory. The graphical plots of Gaussian surround function with the scales of 16, 64 and 128 are shown in Figures 3(a), (b) and (c), respectively.

### 4. Architectures of 2D Gaussian Surround Function

The top level architecture of Gaussian surround function comprises counter, dual port ROM, multiplier, scaler and exponential. The overall architecture of 2D Gaussian surround function is presented in Figure 4. The signal description of complete design is shown in Table 1. The top design is called as "gauss2D" its block diagram consists signals "clk", "reset\_n", "start"

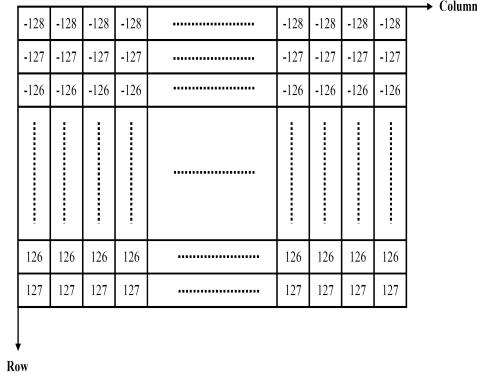


Figure 2. Domain Specified by 'y' Vector Transformed into Array 'y'

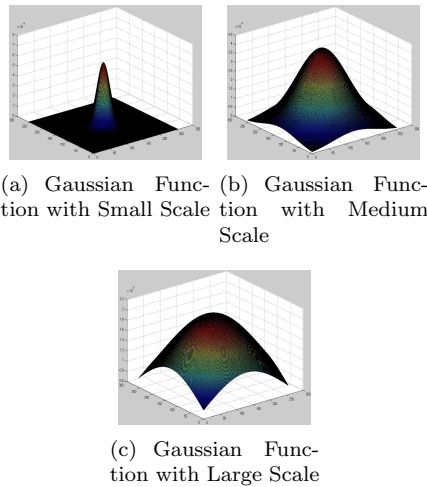


Figure 3. 2D Gaussian Surround Functions with Different Scales

and "gout [7:0]". The signal "gout [7:0]" is the Gaussian output which is produced at every rising edge of "clk" signal. The signal "reset\_n" is the global reset signal which is used to reset the system during power on conditions. The hardware realization of "gauss2D" functional modules in this work is based on adapting enormous amount of pipeline stages mainly to reduce the computation time. In addition to pipelining technique, parallel processing also has been employed to accelerate the Gaussian function. The Table 2, provides the memory specification of the dual port ROM for the Gaussian surround function of size  $256 \times 256$ . The ROM address ranges between 00000000 (corresponds to decimal 0) and 11111111 (corresponds to decimal 255). The contents of the ROM for each location are specified in the second column of Table 2. As is evident from the Table 2, the ROM stores the first row of the spatial co-ordinate 'x' presented in Figure 1. It is possible to produce the array 'x', from the ROM contents since the elements are repeated in the second and subsequent rows of 'x' co-ordinate. Similarly, it is easy to construct the array 'y' presented in Figure 2 by incrementing the address (addr2 [7:0]) of ROM every 256 clock cycle.

Table 2  
Memory Specification for 2D Gaussian Surround Function

Address	Data
00000000	10000000 (-128)
00000001	10000001 (-127)
00000010	10000010 (-126)
00000011	10000011 (-125)
11111110	01111110 (126)
11111111	01111111 (127)

The counters used in the present work is of width 8-bits and is shown in Figure 5. The output of these counters are fed as address input for

Table 1  
Signal Description for the Gaussian Surround Function Design

Signals	Description
clk	This is the global clock signal
reset_n	Active low system reset
start	Asserted to initiate Gaussian function
gout [7:0]	Gaussian Output
enable	Asserted to initiate Counting
cnt_out1 [7:0] and cnt_out2 [7:0]	Counter Outputs
addr1 [7:0] and addr2 [7:0]	Address for Data selection in ROM
dout1 [7:0] and dout2 [7:0]	Output of Dual port ROM
n1 [7:0] and n2 [7:0]	Multiplier inputs of 8-bit
result [15:0]	Multiplier Output of 16-bit

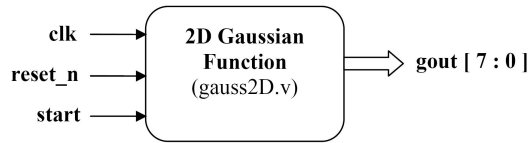


Figure 4. Top Architecture of 2D Gaussian Surround Function

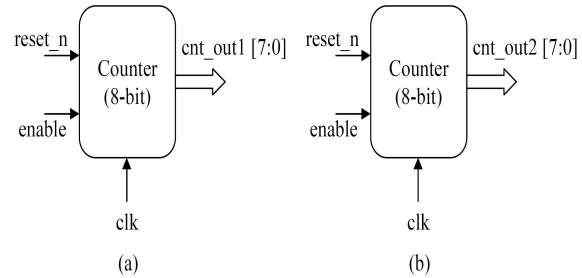


Figure 5. Design of 8-bit Counter with its Output as Address for Dual Port ROM

ROM since the ROM expect the 8-bit address. The ROM address (addr1 [7:0]) increments at every rising edge of clock cycle. However, the other ROM address (addr2 [7:0]) increments while the addr1 reaches its maximum value. This approach produces the matrix 'x' and 'y' as described earlier. The architecture of the dual port ROM is shown in Figure 6 and its signal description is provided in Table 1. The output of ROM generates matrix 'x' and 'y' in a raster scan order.

The multiplier design [6] presented in this work incorporates a high degree of parallel circuits and pipelining of five levels. The multiplier performs the multiplication of two 8-bits unsigned numbers n1 and n2 as shown Figure 7 with its signal description in Table 1. The multiplier of width  $16 \times 16$  may also employed for Gaussian function of larger size. The multiplier result is of width 16-bits. The detailed architecture for the multiplier is shown in Figure 8. The architecture

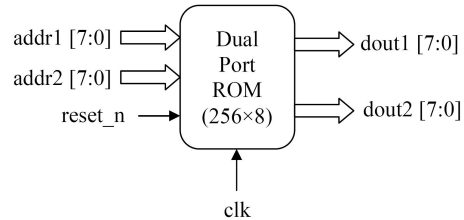


Figure 6. Architecture of Dual Port ROM

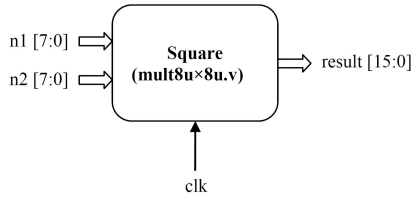
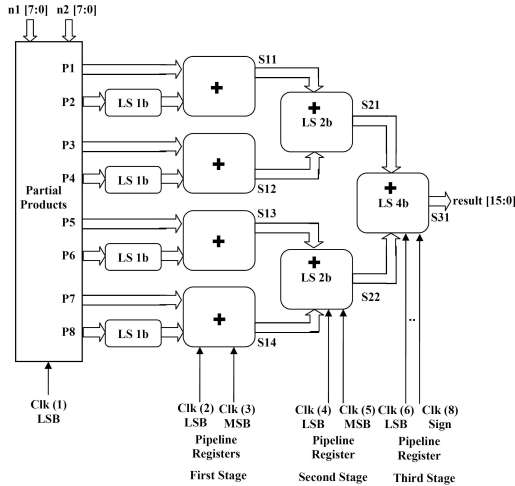
Figure 7. Top Architecture of  $8 \times 8$  Multiplier

Figure 8. Detailed Architecture of Multiplier Design with Eight Pipeline Stages

utilizes many pipelined registers internally. Five pipelined stages are exploited in order to increase the processing speed.

## 5. Experimental Results and Discussions

The proposed FPGA implementation of Gaussian surround function has been coded and tested in Matlab (Version 8.1) first in order to ensure the correct working of the algorithm. Subsequently, the complete system has been coded in Verilog HDL so that it may be implemented on an FPGA or ASIC. The proposed scheme has been coded in

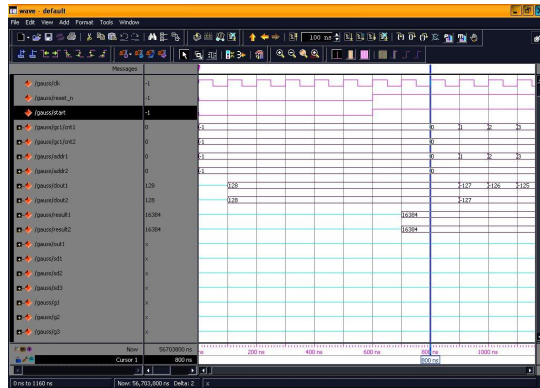
RTL compliant Verilog and the hardware simulation results have been obtained. The system simulation has been done using ModelSim (Version SE 6.4) and Synthesized using Xilinx ISE 9.1i. The algorithm has been implemented on Xilinx Virtex-II XC2VP40-7FG676 FPGA device. In the proposed work, Gaussian design developed is of size  $256 \times 256$  and further can be upgraded to any size without appreciable increase in the hardware. The functional modules comprising of Gaussian control, ROM, Multiplier, Adder and exponential of Gaussian surround function simulated using ModelSim is presented in Figure 9.

The Xilinx generated RTL schematic view of top module of "gauss" is shown in Figure 10. The output of this schematic consists of g1 [7:0], g2 [7:0] and g3 [7:0] which is the Gaussian output with three scales namely, 16, 64 and 128. The detailed schematic produced by this top schematic is presented in Figure 11. There are seven modules of which multiplier, and exponent are replicated. The input to the exponent module is in the range of 0 to 4. Therefore, the exponent module is designed with the look-up table technique instead of designing it.

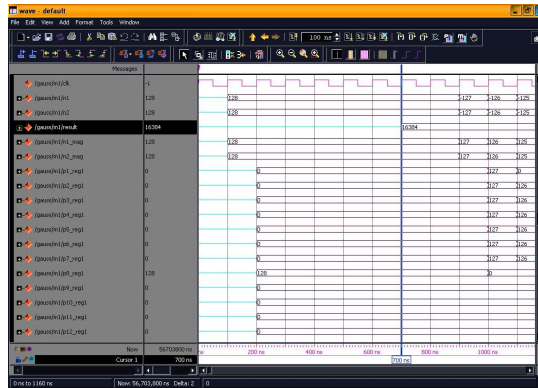
The synthesis as well as place and route results of the Gaussian function is presented in Table 3. The Xilinx place route report for the entire Gaussian function comprising counter, ROM, Multiplier, adder and exponent is shown in Figure 12.

The timing summary for the design as reported by Xilinx ISE tool is as follows:

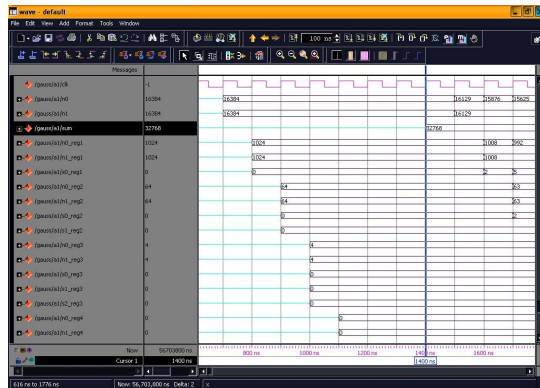
1. Speed Grade : -7
2. Minimum period: 4.483 ns  
(Maximum Frequency: 223.04 MHz)
3. Minimum input arrival time before clock: 1.473 ns
4. Maximum output required time after clock: 6.880 ns
5. Clock period: 4.483 ns  
(frequency: 223.04 MHz)
6. Total number of paths /destination ports: 26126 / 1922
7. Delay: 4.483 ns  
(Levels of Logic = 18)



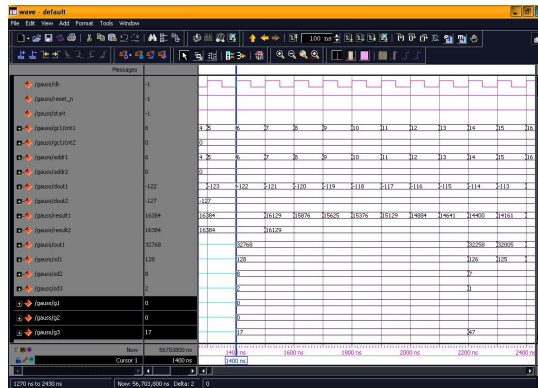
(a) Start of Simulation for 2D Gaussian Surround Function



(b) Waveforms for Multiplier Output



(c) Waveforms for Adder Output



(d) Waveforms for Gaussian Surround Function Output with Three Scales

Figure 9. ModelSim Simulation Waveforms for 2D Gaussian Surround Function

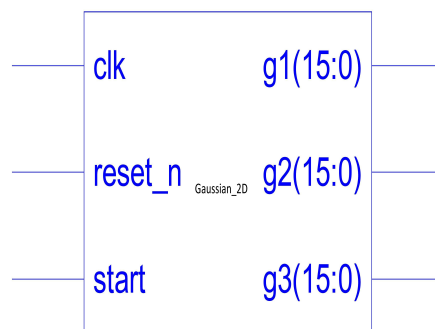


Figure 10. RTL View of the Top Module "gauss"

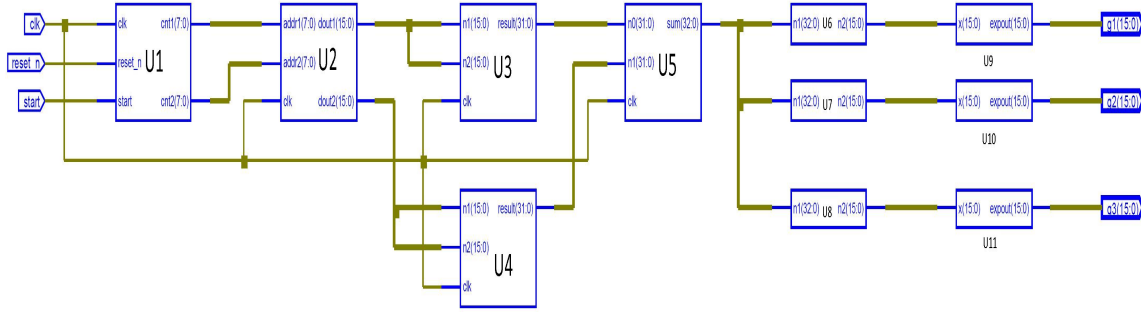


Figure 11. Zoomed View of the Top Module "gauss" **Note:** The module names are not readable in the zoomed view of the Xilinx RTL view. Therefore, in the generated Figure the module names are marked as follows: U1: Gaussian control, U2: Gaussian Memory (ROM), U3 and U4: Multiplier, U5: Adder, U6, U7 and U8: Scale Down Unit, U9, U10 and U11: Exponent

Table 3  
Device Utilization Summary

Selected Device : XC2VP40-7FG676		
Number of Slices	725 out of 19,392	3%
Number of Slice Flip Flops	1224 out of 38,784	3%
Number of 4 input LUTs	825 out of 38,784	2%
Number used as logic	713	
Number used as Shift registers	94	
Number of IOs	51	
Number of bonded IOBs	51 out of 416	12%
Number of GCLKs	1 out of 16	6%

Table 4  
Design Mapped on Various FPGA Devices

Device	LUTs	Gates	Utilization	Frequency
XC2S50-6TQ144	825 out of 1536	62,455	53%	61.992 MHz
XC2S200-6PQ208	825 out of 4704	62,455	17%	61.992 MHz
XC3S1600E-5FG484	825 out of 29504	89,213	5%	160.458 MHz
XC2VP40-7FG676	825 out of 38784	89,213	2%	223.04 MHz



GAUSS2D Project Status			
Project File:	gauss2D.isc	Current State:	Programming File Generated
Module Name:	gauss	• Errors:	No Errors
Target Device:	xc2vp40-7fg676	• Warnings:	<a href="#">192 Warnings</a>
Product Version:	ISE 9.1i	• Updated:	Sun Apr 14 11:22:52 2013

Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	1,224	38,784	3%	
Number of 4 input LUTs	713	38,784	1%	
<b>Logic Distribution</b>				
Number of occupied Slices	725	19,392	3%	
Number of Slices containing only related logic	725	725	100%	
Number of Slices containing unrelated logic	0	725	0%	
<b>Total Number of 4 input LUTs</b>	<b>825</b>	<b>38,784</b>	<b>2%</b>	
Number used as logic	713			
Number used as a route-thru	18			
Number used as Shift registers	94			
Number of bonded IOBs	51	416	12%	
Number of PPC405s	0	2	0%	
Number of Block RAMs	1	192	1%	
Number of GCLKs	1	16	6%	
Number of GTs	0	12	0%	
Number of GT10s	0	0	0%	
<b>Total equivalent gate count for design</b>	<b>89,213</b>			
Additional JTAG gate count for IOBs	2,448			

Figure 12. Device Utilization Summary as Reported by Xilinx Tool

## 6. Conclusion

The 2D Gaussian surround function has been designed for processing high resolution pictures of size  $1600 \times 1200$  at real time rate of 30 frames per second. The Gaussian function design developed is based on the symmetry property which consumes low on-chip memory. The 2D Gaussian function implementation is presented for use in applications such as image enhancement, smoothing, edge detection and filtering etc. The architectures developed was coded in Verilog, conforming to RTL coding guidelines used in industries and fits onto a single chip with a gate count utilization of 89,213 gates. Research work is in progress for employing the Gaussian surround function for Multiscale Retinex based color image enhancement.

## REFERENCES

1. Daniel J. Jobson, Zia-ur Rahman and Glenn A. Woodell, *A Multiscale Retinex for Bridging the Gap Between Color Images and the Human Observation of Scenes*, IEEE Transactions on Image Processing, Vol. 6, No. 7, pp. 965-976, 1997.
2. Glenn Hines, Zia-ur Rahman, Daniel Jobson and Glenn Woodell, *DSP Implementation of the Retinex Image Enhancement Algorithm*, In Defense and Security, International Society for Optics and Photonics, pp. 13-24, 2004.
3. M. C Hanumantharaju, M. Ravishankar, and D. R. Rameshbabu *Design of Novel Algorithm and Architecture for Gaussian Based Color Image Enhancement System for Real Time Applications*, In Proceedings of International Conference on Advances in Computing, Communication, and Control (ICAC3), Vol. 361, pp. 595-608, 2013.
4. Hiroshi Tsutsui, Hideyuki Nakamura, Ryoji Hashimoto, Hiroyuki Okuhata, and Takao Onoye, *An FPGA implementation of real-time retinex video image enhancement*, In IEEE World Automation Congress (WAC), pp. 1-6, 2010.
5. Tsung Han Tsai, Chung-Yuan Lin, and Sz-Yan Li, *Algorithm and Architecture Design of Human-Machine Interaction in Foreground Object Detection with Dynamic Scene*, 2013.
6. Seetharaman Ramachandran, *Digital VLSI*

*Systems Design: A Design Manual for Implementation of Projects on FPGAs and ASICs Using Verilog*, Springer, 2007.

7. Donald G. Bailey, *Design for Embedded Image Processing on FPGAs*, Wiley-IEEE Press, 2011.
8. H. T. Ngo, M.Z. Zhang, L. Tao, and V.K. Asari, *Design of a Digital Architecture for Real-time video, Enhancement based on Illuminance-Reflectance Model*, In Circuits and Systems, 2006. MWSCAS06. 49th IEEE International Midwest Symposium on, volume 1, pages 286290, 2006.
9. L. Tao and V. Asari, *Modified Luminance based MSR for Fast and Efficient Image Enhancement*, In Applied Imagery Pattern Recognition Workshop, 2003, Proceedings. 32nd, pages 174179, 2003.
10. Ajoy K. Ray Tinku Acharya *Image Processing: Principles and Applications*, Wiley-Interscience, 2005.

*M. C Hanumantharaju, et al.,*



**M. C Hanumantharaju** is currently a Associate Professor in the Department of Information Science & Engineering, Dayananda Sagar College of Engineering, Bangalore. He is currently pursuing Ph.D at Visvesvaraya Technological University, Belgaum His research interests includes VLSI Architecture Development for Signal & Image Processing Applications, Synthesis & Optimization of ICs, DSP with FPGAs etc.



**M. T Gopalakrishna** is currently a Associate Professor in the Department of Information Science & Engineering, Dayananda Sagar College of Engineering, Bangalore. He is currently pursuing his Ph.D at Visvesvaraya Technological University, Belgaum. His Research interests includes Digital Image Processing & Computer Vision, Video Surveillance and Document Image Processing.